

PDF Controller

Offer your HTML page for PDF download. Control the conversion with the integrated user interface.



Version: 1.1.0, 2012-02-03

Extension Key: pdfcontroller

Language: en

Keywords: forAdmins, forDevelopers, forAdvanced, pdf, download, html

Copyright 2011-2012, Dirk Wildt - Die Netzmacher, <<http://wildt.at.die-netzmacher.de>>

This document is published under the Open Content License
available from <http://www.opencontent.org/opl.shtml>

The content of this document is related to TYPO3

- a GNU/GPL CMS/Framework available from www.typo3.org

Table of Contents

PDF Controller	1	Backend Tool	22
Screen Shots	3	Defined errors and warnings	24
Frontend	3	Cache	25
Backend	4	Remove the Cache manually	25
Introduction	6	Security	26
What does it do?	6	In principle	26
Manual in PDF	6	html2ps	26
Installation	7	System Requirements	27
Tutorial I	8	Check it by the Plugin!	27
Installation	8	Required	27
Add Pages and Plugins	8	Recommended	27
Configure the Plugin Button	10	Reference	28
Adapt the TypoScript to your needs	10	html2ps	28
TypoScript Snippet	11	tx_pdfcontroller_pi2	29
Tutorial II	12	FAQ	30
Based on	12	CSS compatibility list	30
Page Object	12	Further Information	31
CSS	14	Other extensions published by Die Netzmacher GbR	31
Button for PDF download and for Printing	15	Helpful suggestions	32
Completely TypoScript	17	Forum	32
Optimization	18	Credits	33
Result	18	Darren Gates and Konstantin Bournayev	33
Development	19	Change log	34
Requirements	19	Illustration index	35
Plugin	19		
Frontend Tools	20		

Screen Shots

Frontend

Case I - without configuring any line TypoScript



Illustration 1: Download dialog after the mouse click on the PDF button

You have to add a page with the plugin PDF Controller User Interface and the PDF Controller Button to a page only.

Case II - Website wide PDF downloads and optimized printing

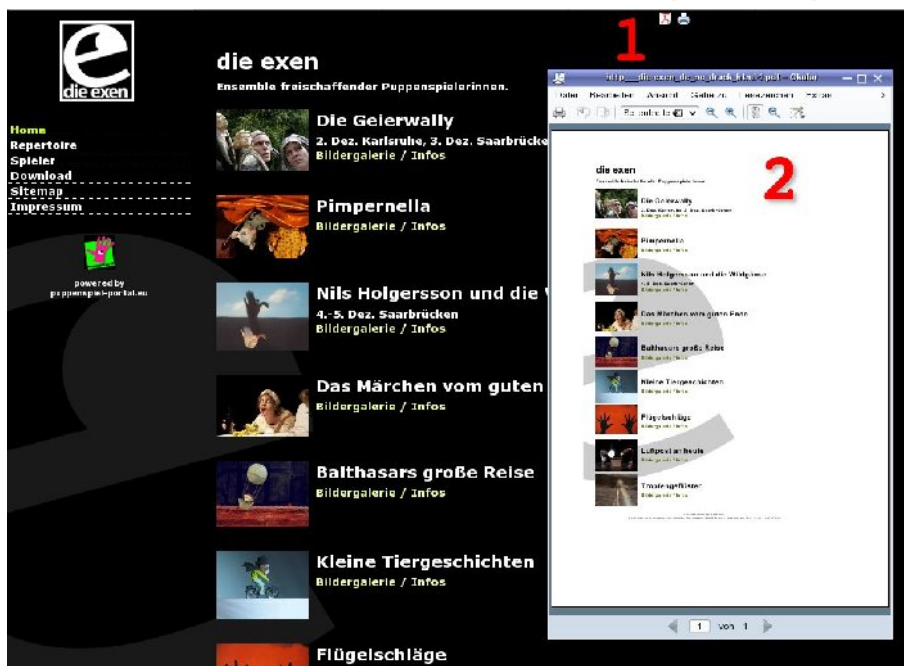
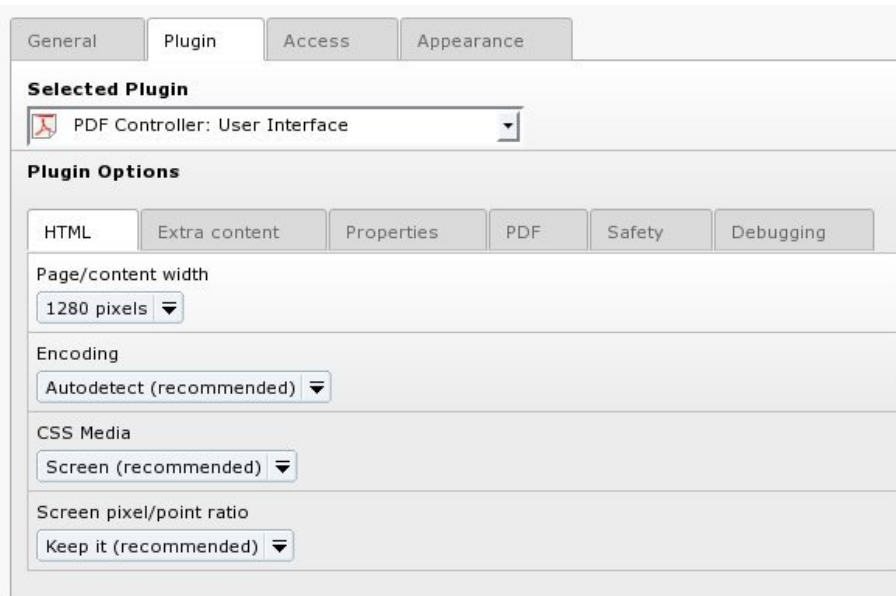


Illustration 2: 1: PDF button and print button generated by the PDF Controller. 2: PDF file in a PDF viewer.

Tutorial II on page 12 below demonstrates, how to integrate the PDF button and the print button (1) website wide. You can configure the conversion, your PDF file (2) can defer to the HTML page (for optimized printing i.e.).

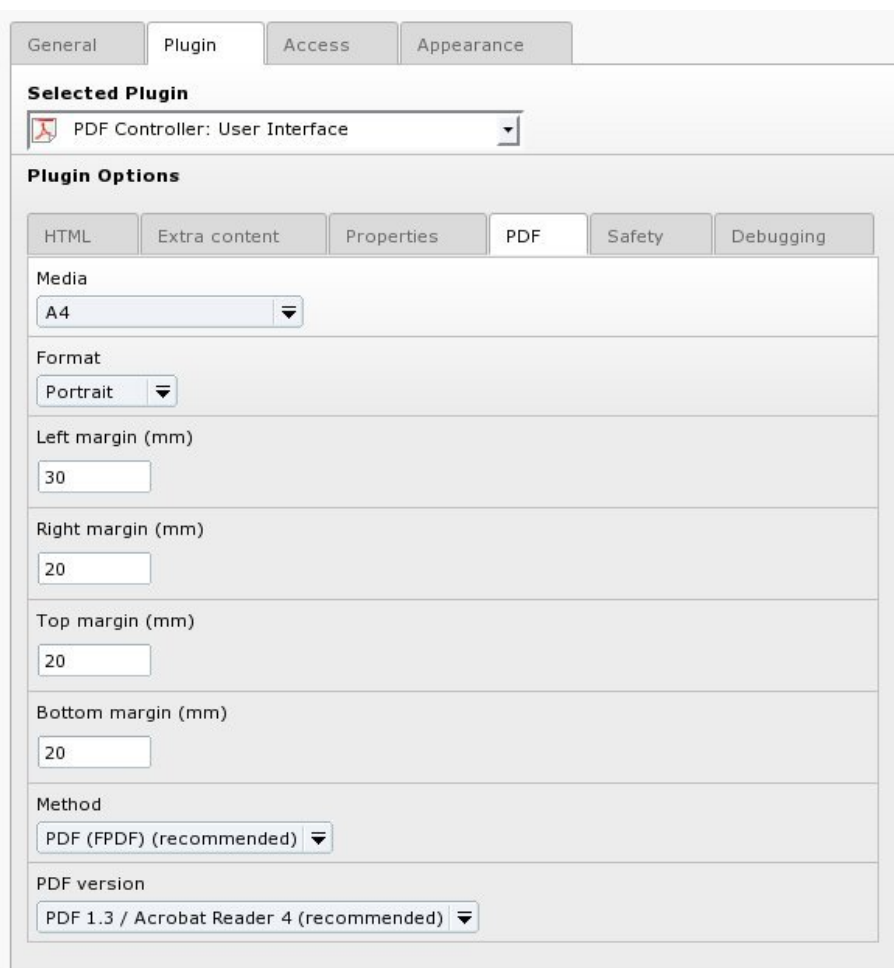
Backend

PDF Controller User Interface



The screenshot shows the 'Plugin' tab of the PDF Controller User Interface. The 'Selected Plugin' is 'PDF Controller: User Interface'. Under 'Plugin Options', the 'HTML' sub-tab is active. The settings include: Page/content width set to 1280 pixels, Encoding set to Autodetect (recommended), CSS Media set to Screen (recommended), and Screen pixel/point ratio set to Keep it (recommended).

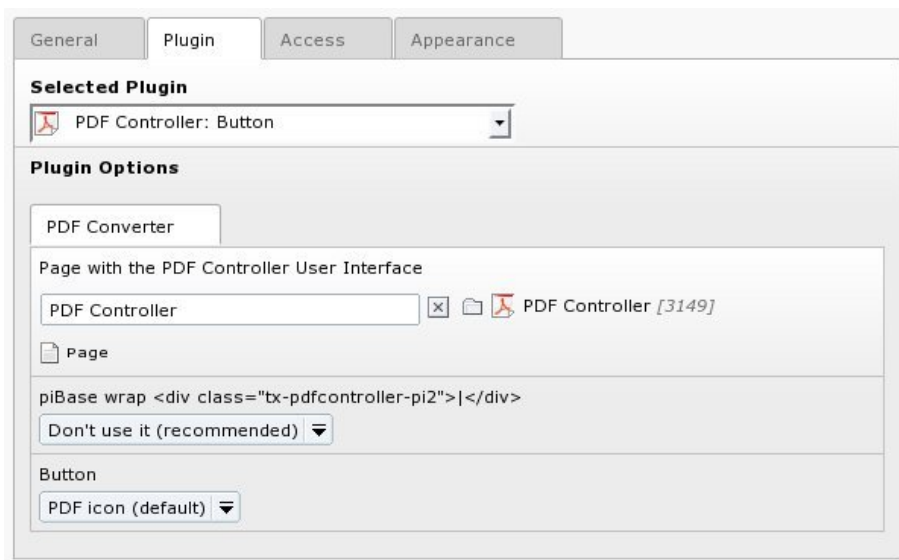
Illustration 3: PDF Controller User Interface: Properties of the HTML page



The screenshot shows the 'Plugin' tab of the PDF Controller User Interface. The 'Selected Plugin' is 'PDF Controller: User Interface'. Under 'Plugin Options', the 'PDF' sub-tab is active. The settings include: Media set to A4, Format set to Portrait, Left margin (mm) set to 30, Right margin (mm) set to 20, Top margin (mm) set to 20, Bottom margin (mm) set to 20, Method set to PDF (FPDF) (recommended), and PDF version set to PDF 1.3 / Acrobat Reader 4 (recommended).

Illustration 4: PDF Controller User Interface: Properties of the PDF document

PDF Controller Button



The screenshot shows a configuration window for the 'PDF Controller Button' plugin. It has four tabs: 'General', 'Plugin', 'Access', and 'Appearance'. The 'Plugin' tab is selected. Under 'Selected Plugin', a dropdown menu shows 'PDF Controller: Button'. Below this, the 'Plugin Options' section contains several settings:

- A 'PDF Converter' tab is visible.
- A section titled 'Page with the PDF Controller User Interface' contains a text input field with 'PDF Controller', a close button (X), a folder icon, and a PDF icon followed by the text 'PDF Controller [3149]'.
- A 'Page' section with a document icon and the text 'Page'.
- A 'piBase wrap' section with a code snippet: `<div class="tx-pdfcontroller-pi2">|</div>`.
- A dropdown menu for the wrap option, currently set to 'Don't use it (recommended)'.
- A 'Button' section with a dropdown menu set to 'PDF icon (default)'.

Illustration 5: PDF Controller Button

Introduction

What does it do?

- The PDF Controller converts your whole HTML page or parts of your HTML page to a PDF document.
- The CSS compatibility is high level (see more information in section "CSS compatibility list" on page 30 below).
- The PDF Controller has an user interface for controlling the PDF process. The user interface is in English or German depending on the language of the backend user.
- The PDF Controller has a second plugin: The PDF Controller Button.
- The PDF Controller has five tools for development (four frontend tools, one backend tool)
- The PDF Controller generates unique file names.
- The PDF Controller doesn't need an extra page object.
- The PDF Controller is secure: You can use it for PDF converting for the current website only by default.
- The PDF Controller has a forum (see Forum on page 32 below).

Manual in PDF

You find this manual as PDF file at

- `doc/manual.pdf`
- http://typo3.org/extensions/repository/view/pdfcontroller/current/info/?tx_terfe_pi1%5BdownloadFile%5D=doc%252Fmanual.pdf

Installation

- Install the extension PDF Controller (pdfcontroller) and PDF Controller Fonts (pdfcontroller_fonts).
- Create in your library folder (or anywhere) the page:
 - PDF Controller
- Add to the page PDF Controller the plugin PDF Controller User Interface
- Add the plugin PDF Controller Button to the page, which should be downloadable as a PDF document.
- Configure the plugin PDF Controller Button: Link to the PDF Controller User Interface.
- Include the static TypeScript template "PDF Controller Button" into the page with the button.

That's all.

Tutorial I

If you like to offer every page for the PDF download, you don't need to add the PDF Controller Button to every page. You define a central PDF Controller Button and integrate it once by TypoScript to every page.

Installation

The installation is the same like in the section Installation on page 7 above. But you don't need to add the PDF Controller Button to a special page.

Add Pages and Plugins

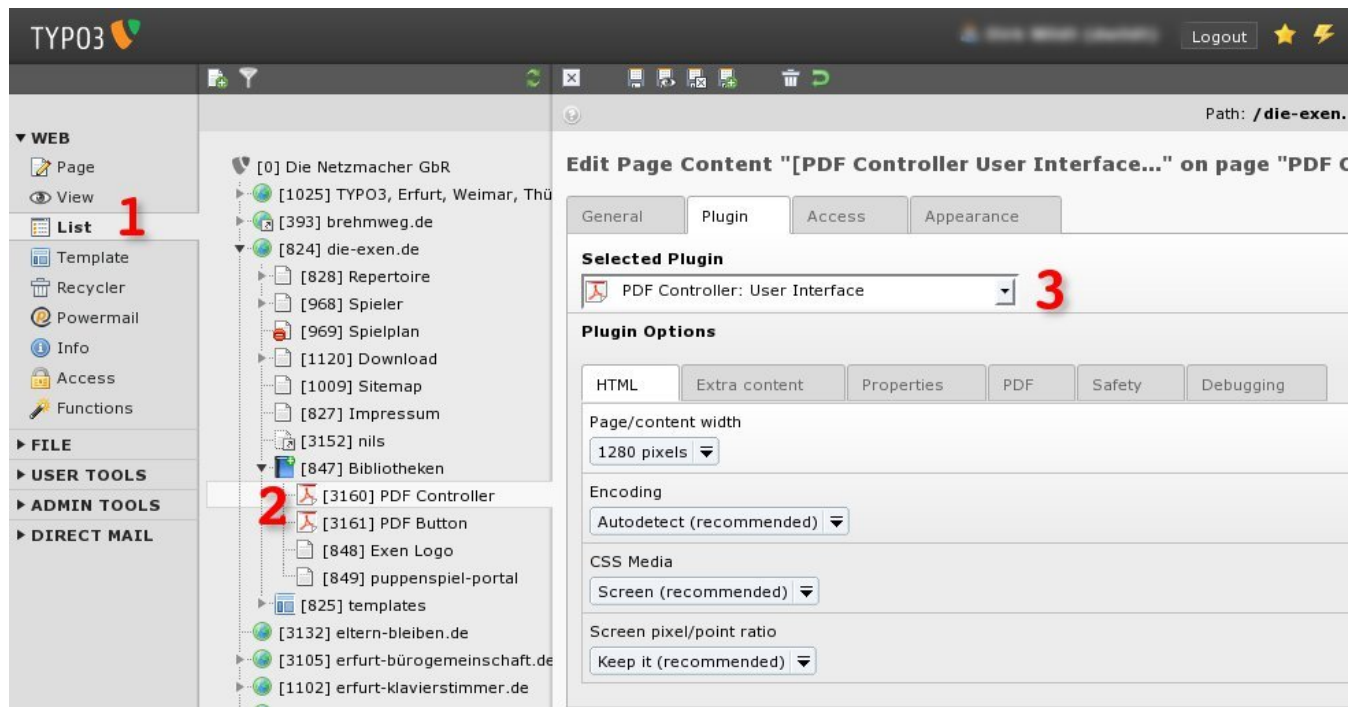


Illustration 6: Pages and plugin

1. Module Web > List
2. Page tree Add to your folder libraries (or another sysfolder) the pages:
 - PDF Controller
 - PDF Button
3. Edit area Add to your pages the correspondingly plugins.
 - page PDF Controller: the plugin PDF Controller User Interface
 - page PDF Button: the plugin PDF Controller Button

Add the page tree icon

If you like the PDF icon for your pages in the page tree like in the illustration below, please configure the page properties of the pages PDF Controller and PDF Button.

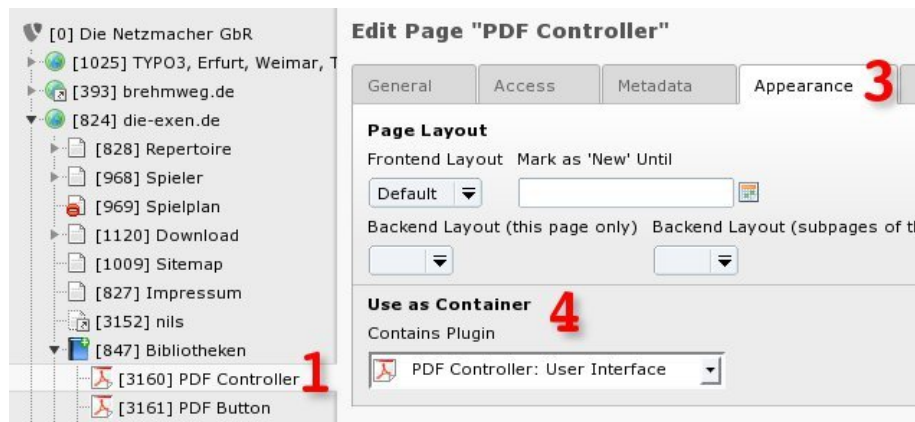


Illustration 7: Page contains the plugin ...

- | | | |
|----|-----------|---|
| 1. | Page tree | PDF Controller |
| 2. | Edit area | Edit page properties |
| 3. | | Tab [Appearance] |
| 4. | | Field [Contains Plugin]: PDF Controller: User Interface |

Repeat the steps for the page PDF Button.

Add more page tree icons?

Do you like the library icon in the illustration above? It's available after installing the extension "TSconfig Pages and Users by extManager" (extkey: tsconf). See "Other extensions published by Die Netzmacher GbR" on page 31 below.

Configure the Plugin Button

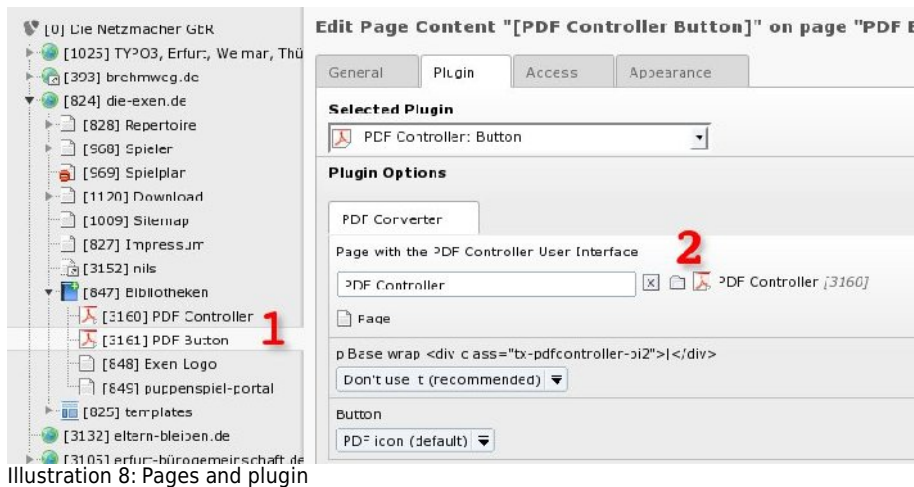


Illustration 8: Pages and plugin

1. Page tree PDF Button
2. Edit area Edit the plugin PDF Controller Button:
Connect the button with the page PDF Controller.

Adapt the TypoScript to your needs

This is an example for a TypoScript without TemplaVoilà. If you are using TemplaVoilà, the workflow will be different. Please inspect your TypoScript code. Please find the TypoScript array, where you want to place the button like in the example below.

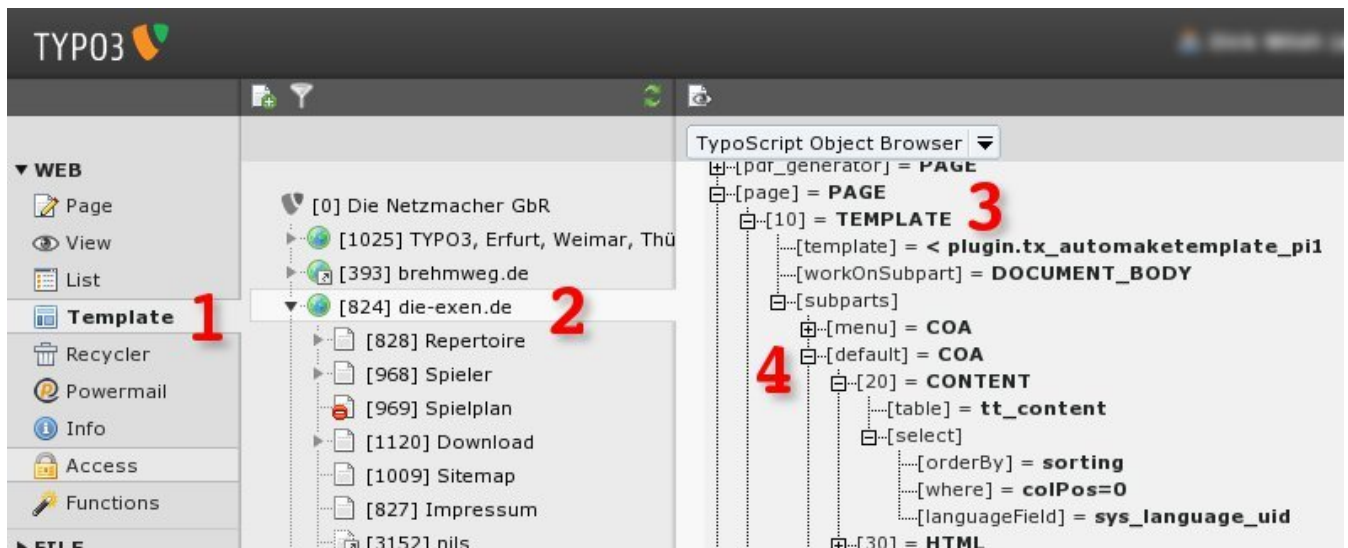


Illustration 9: TypoScript Object Browser

1. Module Web > Template
2. Page tree Root page (in the illustration above: die-exen.de)
3. Edit area TypoScript Object Browser
Look for the content area in your TypoScript page object.
In the illustration above: page.10.template.subparts.default

TypoScript Snippet

The content area is in our example above:

```
page.10.template.subparts.default.20
```

If we want the PDF button on the top of the content area, we have to add something like this:

```
page.10.template.subparts.default.10
```

If we want the PDF button on the bottom of the content area, we have to add something like this:

```
page.10.template.subparts.default.30
```

The whole snippet for our example:

```
page {
  10 {
    subparts {
      default {
        10 = COA
        10 {
          wrap = <div style="padding-top:1em;text-align:right;"> | </div>
          10 < styles.content.get
          10 {
            select {
              // Id of the page PDF Button
              pidInList = 3161
            }
          }
        }
      }
      40 = COA
      40 {
        wrap = <div style="text-align:right;"> | </div>
        10 < styles.content.get
        10 {
          select {
            // Id of the page PDF Button
            pidInList = 3161
          }
        }
      }
    }
  }
}
```

Add Snippet and the Include Static Template

1. Please add the snippet to the TypoScript template at the bottom of the field setup of your root page. This is the quick way. The better way is, to create an extension template with the TypoScript snippet and include the extension template into the TypoScript template of the root page.
2. Please include the static Template "PDF Controller: button (pdfcontroller)" into the TypoScript template of your root-page.

Tutorial II

Usually it is a need, to take another HTML template and another CSS for your PDF document.

Reasons are often:

- You don't need navigation tools like a menu in your PDF document.
- You don't need a header image.
- You like a logo at the top and a footer at the bottom.
- The background color of your HTML page is black (bad for printers).

If you need another HTML template or another CSS or an extended CSS, you have to configure another TYPO3 page object.

Result of Tutorial II is a page object, an optimized HTML page for printing and an optimized PDF document.

Based on

This tutorial is based on

- tutorial I and
- the real example <http://die-exen.de/>

If you didn't follow tutorial I, please do it now.

Your needs will defer from <http://die-exen.de/>. Please adapt the TypoScript snippets below to your needs.

Page Object

Copy the current Page Object

```

1.  //////////////////////////////////////////
2.  //
3.  // Page optimized for printing (typeNum 98)
4.
5.  print < page
6.  print {
7.      typeNum = 98
8.      config >
9.      config {
10.         admPanel          = 0
11.         xhtml_cleaning    = 0
12.         language          = de
13.         locale_all        = de_DE
14.         metaCharset       = UTF-8
15.         doctype           = xhtml_strict
16.         htmlTag_langKey   = de
17.         no_cache          = 1
18.     }
19. }
20. // Page optimized for printing (typeNum 98)

```

- Line 5: Copy your current page object to the new object print. If your current page object isn't called page, please adapt this line to your needs.
- Line 7: typeNum of the print object is 98. 98 is used in context of printing usually. But feel free to take another typeNum.
- Line 8: Remove the config of the page object.
- Line 9-18: Define a new config array adapted to your needs. In the example above the language of the website is German. If it isn't in your case, please adapt localization properties to your needs.
- Line 17: If you are using tt_content only for delivering content, you may cache the print object. Set no_cache to 0. But if you are using records like tt_news or another database, it is recommended not to cache the output.

Remove not needed Properties

Now you have to inspect your print object. It is recommended to inspect the TypoScript with the TypoScript Object Browser.

You have to remove not needed properties. This depends on the configuration of your page object.

In our example - it is based on the real website <http://die-exen.de/> - we have to remove the properties below.

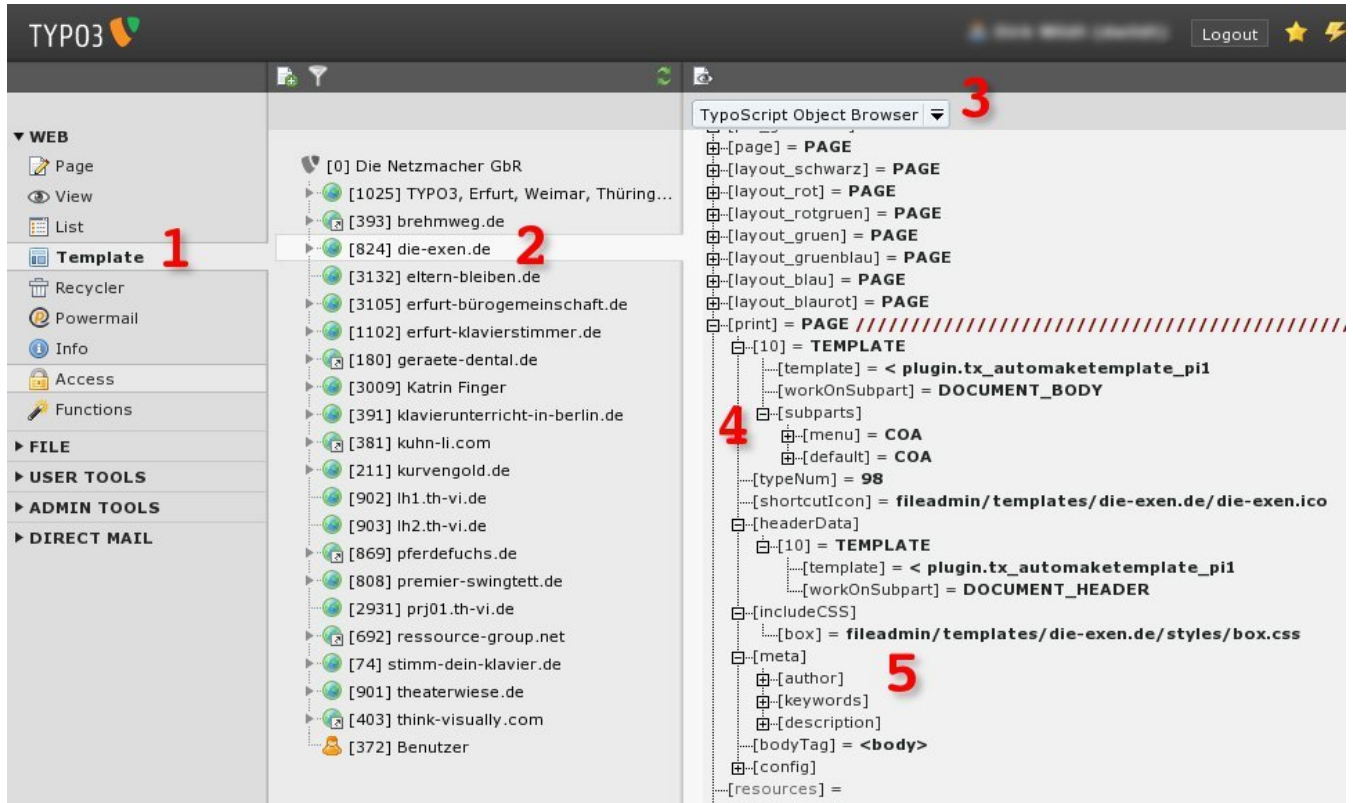


Illustration 10: Inspect your TypoScript

1. Module Web > Template
2. Page tree Root page (in the illustration above: die-exen.de)
3. Edit area TypoScript Object Browser
4. The menu should be removed. It is part of the array subparts.
5. Some meta properties should be removed.
This is for search engines only to avoid double content.

```
// Remove some meta tags
print.meta.keywords >
print.meta.description >
// Replace subpart menu with a space char
print.10.subparts.menu >
print.10.subparts.menu = &nbsp;
// Remove pdf controller button at top and bottom
print.10.subparts.default.10 >
print.10.subparts.default.40 >
```

Add a footer

We need a footer in our example. We add this TypoScript snippet:

```
// Footer
print {
  10 {
    subparts {
      default {
        40 = COA
        40 {
          wrap = <div id="footer_type98">|</div>
          10 = TEXT
          10 {
            wrap = http://die-exen.de|<br />
            typolink {
              parameter                = {page:uid},0
              parameter.insertData    = 1
              returnLast              = url
            }
          }
        }
        21 = TEXT
        21.value = &#169; die-exen.de, c/o puppenspiel-portal.eu, Borntalweg 4, 99092
        Erfurt&nbsp;|&nbsp;
        22 = TEXT
        22.data = date:U
        22.strftime = Gedruckt am %d.%m.%y um %T Uhr
      }
    }
  }
}
```

CSS

We have to overwrite some CSS properties. This is the CSS file:

```
body {
  background-image:url(images/hintergrund_14.gif);
  background-color:white;
  color:black;
}
a,
a:link,
a:visited {
  color: #546600;
}
a:active, a:hover {
  color: #D0FF00;
}
#menu {
  display:none;
}
#footer_type98 {
  font-weight:normal;
  font-size:.6em;
  padding-top:1em;
  text-align:center;
}
```

The TypoScript in prose:

- We need another background colour and background image.
- We need another colour for links
- Don't display the menu
- Some other properties for our footer

Upload the CSS file

We upload the file to the path:

```
fileadmin/templates/die-exen.de/styles/pdf_controller.css
```

Include the CSS file

```
print {
    includeCSS {
        pdf_controller = fileadmin/templates/die-exen.de/styles/pdf_controller.css
    }
}
```

Result

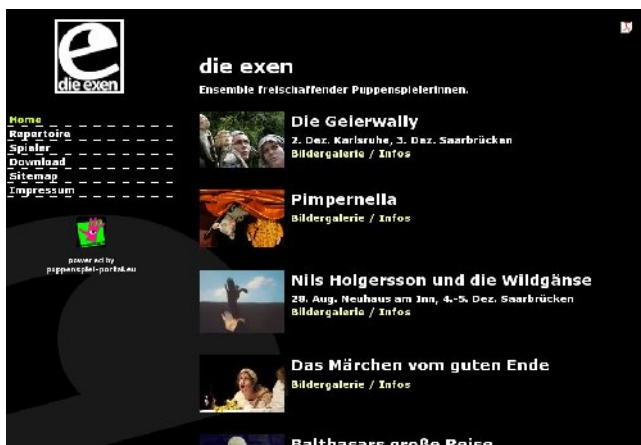


Illustration 11: die-exen.de HTML page



Illustration 12: die-exen.de optimized for printing

The HTML page optimized for printing is a very good base for the PDF conversion.

Next step is, to extend the PDF Controller button with a button for printing.

Button for PDF download and for Printing

The PDF Controller has a master_template for this job.

You have to do this:

- Allocate the master_template to the current button property
- Set the ID of the page PDF Controller in the TYPO3 Constant Editor

Allocate the master_template

```
1. page {
2.     10 {
3.         subparts {
4.             default {
5.                 10 < plugin.tx_pdfcontroller_pi2.master_templates.pdf_and_print_button
6.                 40 < plugin.tx_pdfcontroller_pi2.master_templates.pdf_and_print_button
7.             }
8.         }
9.     }
10. }
```

Line 5 + 6: We allocate the master_template "pdf_and_print_button" to the current button configuration.

Set the ID of the page PDF Controller in the TYPO3 Constant Editor

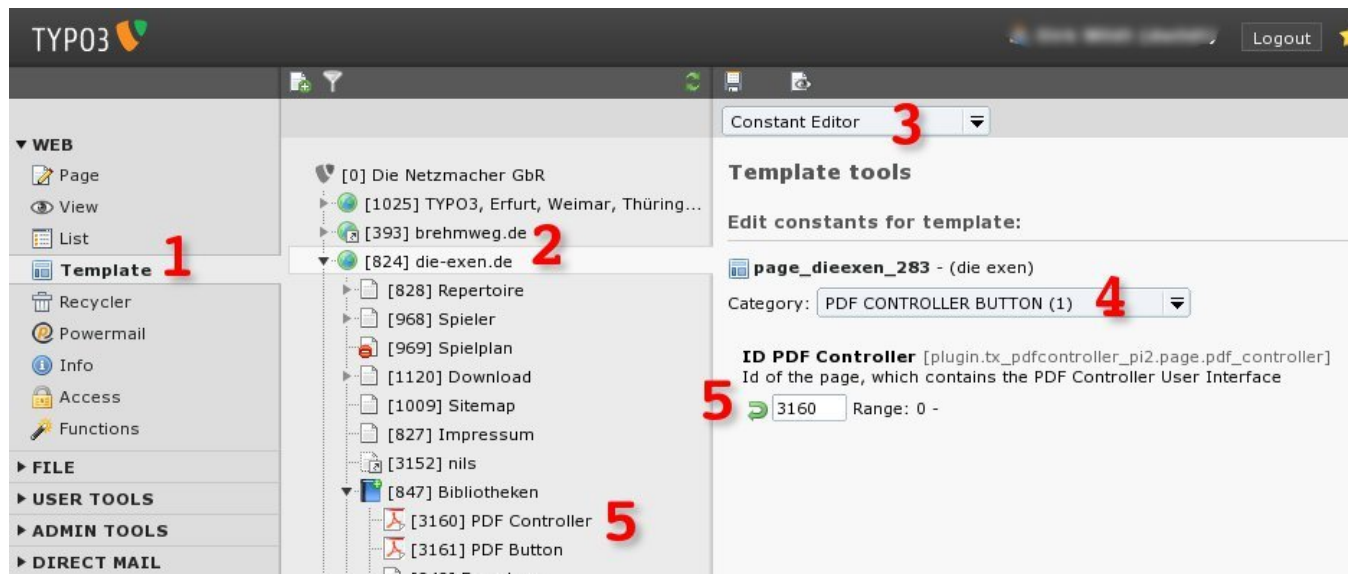


Illustration 13: PDF Controller in the TypoScript Constant Editor

1. Module Web > Template
2. Page tree Root page (in the illustration above: die-exen.de)
3. Edit area TypoScript Constant Editor
4. Category: PDF CONTROLLER BUTTON
5. ID PDF Controller: 3160

Completely TypeScript

This is the completely TypeScript code of Tutorial II (consolidated):

```

page {
  10 {
    subparts {
      default {
        10 < plugin.tx_pdfcontroller_pi2.master_templates.pdf_and_print_button
        40 < plugin.tx_pdfcontroller_pi2.master_templates.pdf_and_print_button
      }
    }
  }
}

print < page
print {
  typeNum = 98
  config >
  config {
    admPanel          = 0
    xhtml_cleaning    = 0
    language          = de
    locale_all        = de_DE
    metaCharset       = UTF-8
    doctype           = xhtml_strict
    htmlTag_langKey   = de
    no_cache          = 1
  }
  meta {
    keywords >
    description >
  }
  includeCSS {
    pdf_controller = fileadmin/templates/die-exen.de/styles/pdf_controller.css
  }
  10 {
    subparts {
      menu >
      menu = &nbsp;
      default {
        10 >
        30 >
        40 >
        40 = COA
        40 {
          wrap = <div id="footer_type98">|</div>
          10 = TEXT
          10 {
            wrap = http://die-exen.de/|<br />
            typolink {
              parameter          = {page:uid},0
              parameter.insertData = 1
              returnLast         = url
            }
          }
          21 = TEXT
          21.value = &#169; die-exen.de, c/o puppenspiel-portal.eu, Borntalweg 4, 99092
Erfurt&nbsp;|&nbsp;
          22 = TEXT
          22.data = date:U
          22.strftime = Gedruckt am %d.%m.%y um %T Uhr
        }
      }
    }
  }
}

```

Optimization

We optimize the PDF file.

Edit the plugin/flexform PDF Controller User Interface on the page PDF Controller:

- Tab [HTML] field [Page/content width]: 640 pixels
- Tab [PDF] fields margins: We reduce the margin

Result

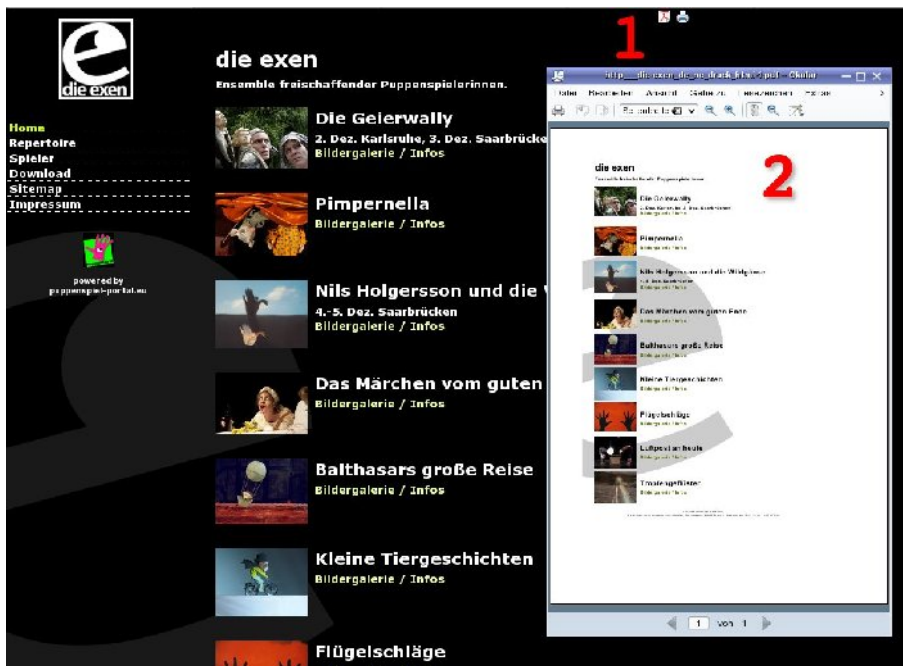


Illustration 14: Result of the tutorial II

1. <http://die-exen.de> with PDF button and print button
2. The generated PDF file in a PDF viewer

Development

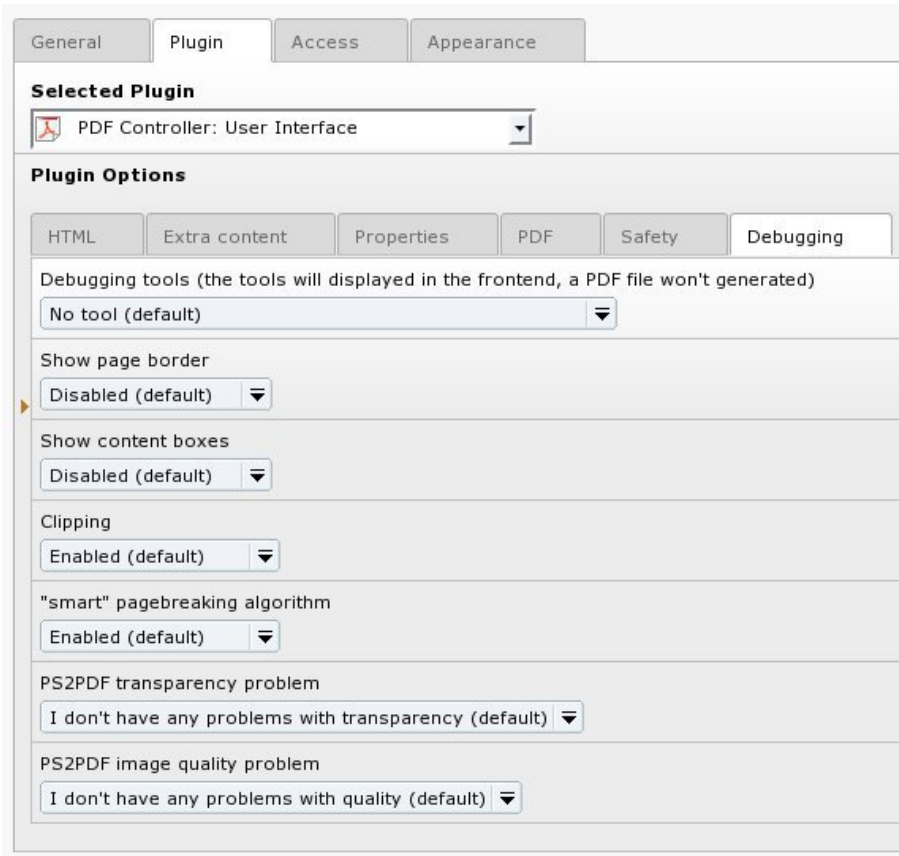
Requirements

You have to be a TYPO3 administrator for using the developer tools.

Plugin

You have some options in the tab [Debugging], if you like to develop the PDF Controller or if you want to get some informations only.

The first select box is powerful: You can select one of four frontend tools. See Frontend Tools on page 20 below.



The screenshot shows the 'Plugin' configuration tab for the PDF Controller. At the top, there are four sub-tabs: 'General', 'Plugin', 'Access', and 'Appearance'. The 'Plugin' tab is active, showing a 'Selected Plugin' dropdown menu with 'PDF Controller: User Interface' selected. Below this is the 'Plugin Options' section, which contains several sub-tabs: 'HTML', 'Extra content', 'Properties', 'PDF', 'Safety', and 'Debugging'. The 'Debugging' sub-tab is selected, displaying various options for debugging the PDF generation process. These options include a dropdown for 'Debugging tools' (set to 'No tool (default)'), checkboxes for 'Show page border' (disabled), 'Show content boxes' (disabled), 'Clipping' (enabled), and 'smart' pagebreaking algorithm (enabled). There are also two more dropdowns for 'PS2PDF transparency problem' and 'PS2PDF image quality problem', both set to 'I don't have any problems with... (default)'.

Illustration 15: PDF Controller with the tab [Debugging]

html2ps documentation

Very helpful documentation, if you want to configure the conversion from HTML to PDF.

html2ps/pdf documentation

1. [FAQ](#)
2. [Requirements and recommendations](#)
3. [Installation notes](#)
4. [Configuring html2ps/pdf](#)
5. [Calling html2ps/pdf](#)
6. [html2ps/pdf HTML directives](#)
7. [Interactive forms](#)
8. Generated content
 - a. [Table of Contents](#)
9. Internals
 - a. [Page breaking algorithm](#)
 - b. [Reserved names](#)
10. API
 - a. [API description](#)
 - b. [How do "fetchers" work?](#)
 - c. [Minimal code samples](#)
 - d. [API events](#)
 - e. [DOM compatibility](#)
11. [HOWTO: Use custom fonts](#)
12. [CSS 2.1 compatibility list](#)
13. [CSS 3 compatibility list](#)

html2ps/pdf (c) TUFA.com

Illustration 18: html2ps documentation

List with parameters, which will send to html2ps

Get a list with the parameters, which will send to html2ps.

List of parameters for html2ps

```
array (
  'smartpagebreak' => true,
  'pixels' => '1280',
  'cssmedia' => 'Screen',
  'scalepoints' => true,
  'renderimages' => '1',
  'renderfields' => '1',
  'renderlinks' => '1',
  'bottommargin' => '20',
  'leftmargin' => '30',
  'media' => 'A4',
  'method' => 'fpdf',
  'pdfversion' => '1.3',
  'rightmargin' => '20',
  'topmargin' => '20',
  'URL' => 'http://die-exen.de/spieler/kathrin-bluechert/',
  'convert' => 'Convert File',
  'process_mode' => 'single',
  'encoding' => '',
  'headerhtml' => '',
  'footerhtml' => '',
  'toc-location' => 'before',
  'pslevel' => '3',
  'output' => '0',
)
```

Full qualified URL

Copy and paste to your browser.

http://die-exen.de/typo3conf/ext/pdfcontroller/res/html2ps_v2043/public_html/demo/html2ps.php?smartpagebreak=1&pixels=1280&cssmedia=Screen&scalepoints=1&renderimages=1&renderfields=1&renderlinks=1&bottommargin=20&leftmargin=30&media=A4&method=fpdf&pdfversion=1.3&rightmargin=20&topmargin=20&URL=http://die-exen.de/spieler/kathrin-bluechert/&convert=Convert File&process_mode=single&encoding=&headerhtml=&footerhtml=&toc-location=before&pslevel=3&output=0

Illustration 19: Report with parameters, which will send to html2ps

Backend Tool

Enable DRS - the Development Reporting System

The PDF Controller is supported by the DRS - the Development Reporting System. You enable the DRS with the extension manager of the PDF Controller.

You can follow the complete work flow of the plugin PDF Controller.

If DRS is enabled, you will receive a report in the backend. It is supporting you in case of errors or if you like to develop the PDF Controller.

You need the extension devlog.

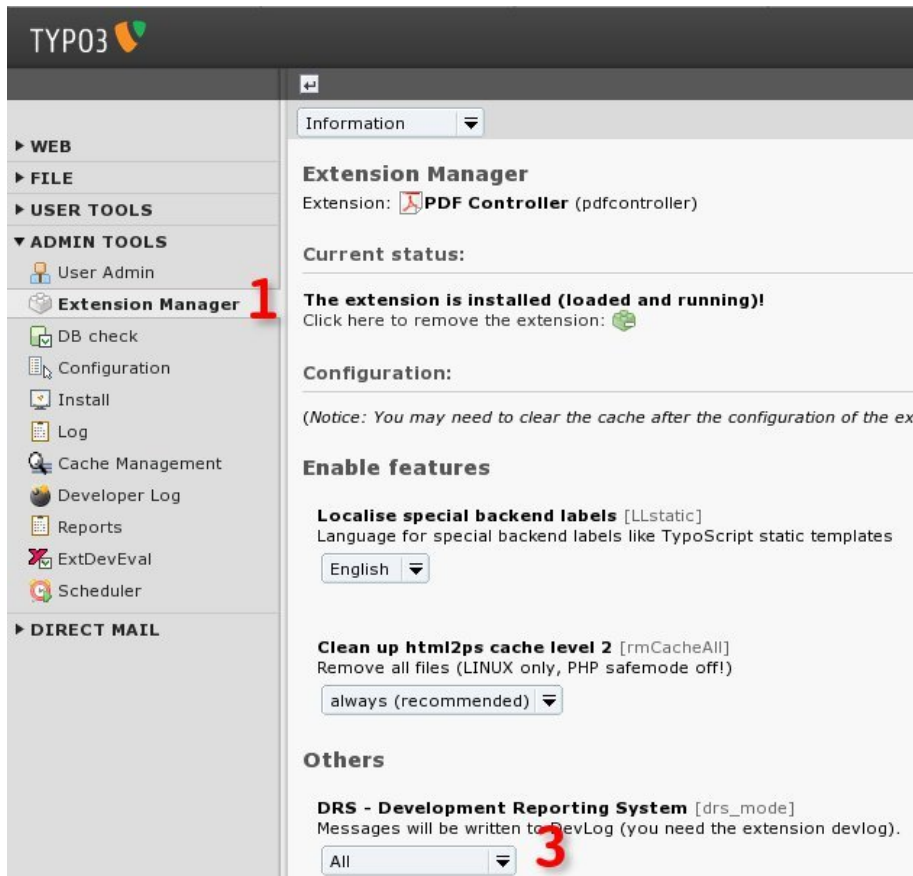


Illustration 20: PDF Controller in the extension manager

1. Module Admin Tools > Extension Manager
2. Edit area Select PDF Controller (pdfcontroller)
3. Enable the DRS

Get the Report

[illegible]

Illustration 21: DRS report

1. Module Admin Tools > Developer Log

Defined errors and warnings

Warning #1: piVars aren't set

The page with the plugin PDF Controller is called without any parameter.

Warning #2: piVars[URL] isn't set

The page with the plugin PDF Controller is called without the parameter &tx_pdfcontroller_pi1[URL].

Warning #3: piVars[URL] is empty

The page with the plugin PDF Controller is called with the parameter &tx_pdfcontroller_pi1[URL] but the parameter is empty.

Cache

Remove the Cache manually

html2ps caches the rendered PDF documents.

It is recommended to clean up the cache from time to time.

Cached files are stored in:

- `ext/pdfcontroller/res/html2ps_v2043/public_html/cache/`

Security

In principle

The external php code of html2ps can be used only within TYPO3. If there is an external call - a call from another website - code won't be executed. The result will be an error message.

html2ps

The html2ps module has a security restriction in particular:

- You can use it for a conversion to PDF for URL of the current website only.

If you have the need to use the core for conversion to PDF of an external URL, please contact the developers. See "Forum" on page 32 below.

System Requirements

PDF Controller is using the PHP project html2ps (see Credits on page 33 below).

Check it by the Plugin!

You can check your system supported by the plugin PDF Controller User Interface. See "Check system requirements" on page 20 above.

Required

- PHP 4.1.0 or newer (PHP 5 supported too)
- native_OR_Active-Link DOM XML extension installed

Recommended

- 'gzlib' PHP extension
- 'iconv' PHP extension

Reference

html2ps

html2ps isn't controlled by TypoScript properties. You control html2ps with URL parameters.

You configure the parameters in the plugin/flexform PDF Controller User Interface.

But you can configure missing parameters and – if you need another value than in the flexform/plugin – all parameters of the flexform/plugin by sending this parameter in the URL.

html2ps/pdf script parameters

The html2ps/pdf script parameters are well documented by html2ps.

Please call the html2ps/pdf documentation (see section html2ps documentation on page 21 above). Click the link:

- [Calling html2ps/pdf](#)

Example

You have set the HTML page width to 640 pixels in the flexform/plugin. But there is one page with a different layout. You need 900 pixels width.

- The html2ps parameter is "pixels"
- The parameter for the URL is than tx_pdfcontroller_pi1[pixels]

Workflow:

- Inspect your TypoScript with the TypoScript Object Browser on the page, where you needed 900 pixels.
- Look for the array, which is configuring your PDF button.
- Add to the value of the additional parameters your parameter: &tx_pdfcontroller_pi1[pixels]=900
- If it is the first parameter behind the pipe, you need a double "&" probably.

```
page.10.subparts.default.10.10.imageLinkWrap.typolink.additionalParams.wrap = \
    &tx_pdfcontroller_pi1[URL]={getIndpEnv:TYPO3_SITE_URL}|&&tx_pdfcontroller_pi1[pixels]=900
```

tx_pdfcontroller_pi2

Property:	Data type:	Description:
button	TCO	<p>TypoScript for rendering the PDF download button. The used markers are replaced only in this TypoScript. If you are change the TypoScript you have to replace the markers manually.</p> <p>Example: If you like to use both - a pdf button and a print button - you can replace the TypoScript like in the example "Allocate the master_template " on page 15 above.</p> <p>Default code:</p> <pre> plugin.tx_pdfcontroller_pi2 { button = IMAGE button { wrap = &nbsp; file = ###IMAGEFILE### altText = TEXT altText { value = Download content as PDF file lang { de = Inhalt als PDF-Datei herunterladen } } titleText < .altText imageLinkWrap = 1 imageLinkWrap { enable = 1 typolink { parameter = ###PARAMETER### additionalParams = ###ADDITIONALPARAMS### } } } } </pre>
master_templates	array	<p>If you like to use both - a pdf button and a print button - you can allocate a master template.</p> <p>Example: See "Allocate the master_template " on page 15 above.</p> <p>Default code:</p> <pre> plugin.tx_pdfcontroller_pi2 { master_templates { pdf_and_print_button = COA pdf_and_print_button { // Please inspect the code with // the TypoScript Object Browser } } } </pre>
_CSS_DEFAULT_STYLE	array	<p>Inline CSS. It will included while runtime.</p> <p>Default code:</p> <pre> plugin.tx_pdfcontroller_pi2 { _CSS_DEFAULT_STYLE (.tx-pdfcontroller-pi2 { text-align: right; } .tx-pdfcontroller-pi2 ul { font-weight: bold; list-style-type: none; margin: 0; padding: 0; } .tx-pdfcontroller-pi2 ul li { display:inline; margin: 0; padding-left:.6em; })} </pre>

FAQ

CSS compatibility list

CSS 2.1

Please call the html2ps/pdf documentation (see section html2ps documentation on page 21 above). Click the link:

- [CSS 2.1 compatibility list](#)


CSS 3

Please call the html2ps/pdf documentation (see section html2ps documentation on page 21 above). Click the link:

- [CSS 3 compatibility list](#)

Further Information

Other extensions published by Die Netzmacher GbR

-  Browser - the TYPO3-Frontend-Engine. The Browser speeds up the developing of TYPO3 extension development eight times! The Browser is the fast way for your data to the TYPO3 frontend. It displays content from related tables. You need 1 line TypoScript for a result list with a search form, a record browser and an a-z browser. Images are wrapped self-acting. SEO, Search Engine Optimization, is integrated.
<http://typo3.org/extensions/repository/view/browser/current/>
-  Green Cars (Grüne Autos) - Database optimized for ecological cars. It is a case study and demonstrates, how to get a complex database with the Browser in three hours only.
http://typo3.org/extensions/repository/view/green_cars/current/
-  Jobmarket is a catalogue with job offers. Views, the a-z-browser, the page-browser, the search, social bookmarks and a lot of other stuff can be configured by the Browser plugin with the mouse. Job Market hasn't any PHP code, it should be easy to adapt it to your needs.
http://typo3.org/extensions/repository/view/job_market/current/
-  Juridat - Database for Juridical Data. Juridat provides a juridical data base with backend functionality. Juridat is out of the box. It is a teamwork with the Browser.
<http://typo3.org/extensions/repository/view/juridat/current/>
-  logical_form: A very small frontend plugin for evaluating forms. I.e. you can evaluate mail addresses.
http://typo3.org/extensions/repository/view/logical_form/current/
-  Majordomo: For subscribing to and unsubscribing from a majordomo mailing list.
<http://typo3.org/extensions/repository/view/majordomo/current/>
-  Organiser provides a lot of features for handling news, events, staff, headquarters, locations, repertoire, workshops and a calendar. Sell your online tickets! Organiser is based on the Browser.
<http://typo3-organiser.de/>
-  Quick Shop (extkey: quick_shop): The fastest shop in the history of TYPO3. Install it with one mouse click only! Quick Shop is based on the Browser and powermail.
http://typo3.org/extensions/repository/view/quick_shop/current/
-  seo_dynamic_tag: Search Engine Optimisation. You can generate values dynamically with this extension especially for the <title>-tag, for the <meta>-tag description and the <meta>-tag keywords.
http://typo3.org/extensions/repository/view/seo_dynamic_tag/current/
-  TSconfig Pages and Users by extManager (extkey: tsconf): Configure the eight most commonly used TSconfig properties with the mouse - like page tree uids, activated extended view, activated clipboard, ...
<http://typo3.org/extensions/repository/view/tsconf/current/>
-  tt_news select configuration (extkey: ttnews_selectconf) enables to select tt_news by any SQL clause. The extension adds an andWhere clause to the SQL query of the tt_news plugin. I.e. it is possible to display tt_news items in dependence on the ownership of a fe_user.
http://typo3.org/extensions/repository/view/ttnews_selectconf/current/
-  Wine Catalogue provides a data base for wine with regions, wineries, styles, variety and ageing among others. It is localized. English, German and Spanish is out of the box. Wine based on the extension Browser.
<http://typo3.org/extensions/repository/view/wine/current/>

Helpful suggestions

Forum

If you have helpful suggestions, feel free to publish any question, bug or code snippet on

- <http://typo3-browser-forum.de/>

Look for the forum TYPO3 PDF Controller.

Posts are welcome in English and German.

Credits

Darren Gates and Konstantin Bournayev

To Darren Gates and Konstantin Bournayev, who developed html2ps. If you like to donate to the html2ps project, visit this URL

- http://www.tufat.com/s_html2ps_html2pdf.htm

You will find a link for donation on the top of the page.

Change log

1.1.0 **Security fix**

1.0.1 **Improvements**

- * #32516: Systemcheck should check fonts in directory of pdfcontroller_fonts
- * #32513: typeNum print template is configurable by the TypoScript constant editor
- * #32482: PDF Controller suggests to install Fonts

Bugfixes

- * #32518: Wrong field name for proxy in the flexform. Thanks to Borries Jensen
- * #31191: Missing folders temp and out. Thanks to Borries Jensen
- * #31190: Parameter are missing - without RealURL only. Thanks to Borries Jensen

1.0.0 **Initial release**

Illustration index

Illustration 1: Download dialog after the mouse click on the PDF button	3
Illustration 2: 1: PDF button and print button generated by the PDF Controller. 2: PDF file in a PDF viewer.	3
Illustration 3: PDF Controller User Interface: Properties of the HTML page	4
Illustration 4: PDF Controller User Interface: Properties of the PDF document	4
Illustration 5: PDF Controller Button	5
Illustration 6: Pages and plugin	8
Illustration 7: Page contains the plugin	9
Illustration 8: Pages and plugin	10
Illustration 9: TypeScript Object Browser	10
Illustration 10: Inspect your TypeScript	13
Illustration 11: die-exen.de HTML page	15
Illustration 12: die-exen.de optimized for printing	15
Illustration 13: PDF Controller in the TypeScript Constant Editor	16
Illustration 14: Result of the tutorial II	18
Illustration 15: PDF Controller with the tab [Debugging]	19
Illustration 16: Report with checked system requirements	20
Illustration 17: Form for sending an URL manually	20
Illustration 18: html2ps documentation	21
Illustration 19: Report with parameters, which will send to html2ps	21
Illustration 20: PDF Controller in the extension manager	22
Illustration 21: DRS report	23